МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів
Кафедра систем управління літальних апаратів

Лабораторна робота № 11

з дисципліни «Алгоритмізація та програмування»
Тема: "Розробка десктоп-застосунків в середовищі QtCreator"

ХАІ.301.141.319а. 27 ЛР

Виконав студент гр. 319а
Владислав РУДНЄВ
(Підпис,дата) (П.І.Б.)

Перевірив к.т.н., доц.
(вчена ступінь, вчене звання)
Олена ГАВРИЛЕНКО
(підпис, дата) (П.І.Б.)

2024

МЕТА РОБОТИ

Ознайомитися з основами розробки програм з використанням QtDesigner і навчитися розробляти десктоп-застосунки із графічним користувацьким інтерфейсом для введення/виведення даних на мові програмування C++ в середовищі QtCreator.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Вивчити алгоритм створення проекту Qt Widgets Application в середовищі QtCreator. Ознайомитись з налаштуваннями основний елементів для введення, виведення, компоновки форми і управління.

Завдання 2. Для вирішення завдання (табл.1) відповідно до варіанта:

QT_task53.Вирішити лінійне рівняння $A \cdot x + B = 0$, задане своїми коефіцієнтами A і B (коефіцієнт A НЕ дорівнює 0).

A. Спроектувати і реалізувати в конструкторі форм графічний інтерфейс програми з віджетами QLabel, QLineEdit і QPushButton. *Використати додаткові віджети: QPushButton, QToolBox, QDoubleSpinBox, QTreeWidget, QTextBrowser, QQuickWidget

B. Додати і відлагодити програмний код для введення вхідних даних з перевіркою на коректність (використати QMessageBox для виведення сповіщень), обчислень і виведення результатів.

C*. Додати пункти меню у QMenuBar для зчитування вхідних даних і збереження результатів в файл з використанням стандартних діалогів для вибору файла.

ВИКОНАННЯ РОБОТИ

Завдання 1.

Вирішення задачі QT_task53.

Вхідні дані:

- A — коефіцієнт при x у рівнянні,
  - тип: double, обмеження:
  - A не дорівнює 0

- B — вільний член рівняння,
  - тип: double,
  - обмеження: будь-яке дійсне число

Вихідні дані:

- x — розв’язок рівняння $A \cdot x + B = 0$,
  - тип: double

- error_msg — повідомлення про помилку у випадку некоректних вхідних даних (наприклад, якщо A = 0),
  - тип: QString

Алгоритм:

1. Зчитати вхідні значення коефіцієнтів A і B.
2. Перевірити, чи A не дорівнює 0:
   - Якщо A = 0, вивести повідомлення про помилку (наприклад, "Коефіцієнт A не повинен дорівнювати 0!") і припинити обчислення.
   - Інакше перейти до кроку 3.
3. Обчислити розв’язок рівняння за формулою:
   x = -B / A
4. Вивести отримане значення x як результат.

Лістинг коду дод. А (стр.5 — 9)

Скрін-шоти вікна виконання програми та десктоп-застосунку дод. Б (сторінка 10 — 14)

ВИСНОВОК

У результаті виконання роботи було досягнуто мету — опановано базові принципи розробки графічного інтерфейсу з використанням Qt Designer та мови C++ у середовищі Qt Creator. Створено десктоп-застосунок для введення, обробки та виведення даних. Отримано практичні навички роботи з віджетами, сигналами й слотами, що є основою для створення повноцінних GUI-програм.

ДОДАТОК А
Лістинінг коду та опис алгоритму створення

№mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

#include <QFileDialog>          // Для відкриття діалогів вибору файлу
#include <QFile>                // Для роботи з файлами
#include <QTextStream>          // Для зручного читання і запису тексту у файл
#include <QMessageBox>          // Для показу повідомлень користувачу
#include <QRegularExpression>   // Для роботи з регулярними виразами

// Конструктор класу MainWindow
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);  // Ініціалізація UI із файлу .ui

    // Підключення сигналів до відповідних слотів:
    // При натисканні кнопки обчислити — викликається функція обчислення
    connect(ui->pushButton_calc, &QPushButton::clicked, this, &MainWindow::on_pushButton_calc_clicked);

    // Пункти меню "Відкрити" та "Зберегти" відкривають відповідні методи
    connect(ui->actionOpen, &QAction::triggered, this, &MainWindow::openFile);
    connect(ui->actionSave, &QAction::triggered, this, &MainWindow::saveFile);

    // Пункт меню "Вихід" завершує роботу програми
    connect(ui->actionExit, &QAction::triggered, qApp, &QApplication::quit);
}

// Деструктор класу для очищення пам'яті
MainWindow::~MainWindow()
{
    delete ui;
}

// Обробка натискання кнопки "Обчислити"
void MainWindow::on_pushButton_calc_clicked()
{
    double a = ui->doubleSpinBox_A->value();  // Зчитуємо коефіцієнт A з відповідного віджета
    double b = ui->doubleSpinBox_B->value();  // Зчитуємо коефіцієнт B

    // Перевірка, що A не дорівнює нулю
    if (a == 0.0) {
        QMessageBox::warning(this, "Помилка", "Коефіцієнт A не повинен дорівнювати 0!");
```

```
        return; // Перериваємо виконання, бо розв’язок не визначений
    }

    // Обчислення розв’язку рівняння A * x + B = 0
    double x = -b / a;
    QString resultText = QString("Розв’язок: x = %1").arg(x);

    // Виводимо результат у відповідний QLabel
    ui->label_result->setText(resultText);

    // Додаємо запис у історію обчислень
    addToHistory(a, b, x);
}

// Функція додавання обчислень до списку історії
void MainWindow::addToHistory(double a, double b, double result)
{
    QString entry = QString("A=%1, B=%2 → x=%3").arg(a).arg(b).arg(result);
    ui->listWidget_history->addItem(entry);
}

// Відкриття файлу та зчитування даних для A і B
void MainWindow::openFile()
{
    // Відкриваємо діалог вибору файлу для відкриття
     QString fileName = QFileDialog::getOpenFileName(this, "Відкрити файл", "", "Text Files (*.txt);;All Files
(*)");
    if (fileName.isEmpty())
        return;  // Якщо файл не обрали — повертаємось

    QFile file(fileName);
    if (!file.open(QIODevice::ReadOnly | QIODevice::Text)) {
        QMessageBox::warning(this, "Помилка", "Не вдалося відкрити файл!");
        return;
    }

    // Зчитуємо весь вміст файлу у рядок
    QTextStream in(&file);
    QString content = in.readAll();
    file.close();

    // Виводимо вміст файлу у текстовий браузер (для перегляду)
    ui->textBrowser->setText(content);

    // Припускаємо, що у файлі є коефіцієнти у форматі: "A 2.5 B 3.7"
    // Розділяємо текст за пробілами або іншими роздільниками
    QStringList parts = content.split(QRegularExpression("\\s+"), Qt::SkipEmptyParts);
```

```
    double a = 0, b = 0;
    bool aOk = false, bOk = false;

    // Ітеруємо по списку частин, шукаємо ключі "A" та "B" та їхні значення
    for (int i = 0; i < parts.size(); ++i) {
        if (parts[i] == "A" && i + 1 < parts.size()) {
            a = parts[i + 1].toDouble(&aOk);
            i++;  // Пропускаємо значення, бо вже його прочитали
        } else if (parts[i] == "B" && i + 1 < parts.size()) {
            b = parts[i + 1].toDouble(&bOk);
            i++;
        }
    }

    // Якщо обидва коефіцієнти успішно зчитані, встановлюємо їх у віджети
    if (aOk && bOk) {
        ui->doubleSpinBox_A->setValue(a);
        ui->doubleSpinBox_B->setValue(b);
    } else {
        QMessageBox::warning(this, "Помилка", "Невірний формат файлу або відсутні коефіцієнти A і B.");
    }
}

// Збереження результатів і історії у файл
void MainWindow::saveFile()
{
    // Відкриваємо діалог збереження файлу
     QString fileName = QFileDialog::getSaveFileName(this, "Зберегти файл", "", "Text Files (*.txt);;All Files
(*)");
    if (fileName.isEmpty())
        return;  // Якщо не вибрали файл — вихід

    QFile file(fileName);
    if (!file.open(QIODevice::WriteOnly | QIODevice::Text)) {
        QMessageBox::warning(this, "Помилка", "Не вдалося зберегти файл!");
        return;
    }

    QTextStream out(&file);

    // Записуємо коефіцієнти у файл у зручному форматі
    out << "A " << ui->doubleSpinBox_A->value() << "\n";
    out << "B " << ui->doubleSpinBox_B->value() << "\n";

    // Записуємо результат розв’язку
    out << ui->label_result->text() << "\n";

    // Записуємо усю історію обчислень рядок за рядком
```

```
    for (int i = 0; i < ui->listWidget_history->count(); ++i) {
        out << ui->listWidget_history->item(i)->text() << "\n";
    }

    file.close();
}
```

№mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

// Простір імен Qt для користувацького інтерфейсу,
// який генерується Qt Designer (файл mainwindow.ui)
QT_BEGIN_NAMESPACE
namespace Ui {
class MainWindow;  // Клас, що відповідає за UI (інтерфейс користувача)
}
QT_END_NAMESPACE

// Клас головного вікна програми, успадковує QMainWindow
class MainWindow : public QMainWindow
{
    Q_OBJECT  // Макрос, який необхідний для роботи сигналів та слотів Qt

public:
    // Конструктор класу, приймає вказівник на батьківський віджет
    explicit MainWindow(QWidget *parent = nullptr);

    // Деструктор класу для звільнення ресурсів
    ~MainWindow();

private slots:
    // Слот, що викликається при натисканні кнопки "Обчислити"
    void on_pushButton_calc_clicked();

    // Слот для відкриття файлу через діалог та завантаження даних
    void openFile();

    // Слот для збереження результатів у файл через діалог
    void saveFile();

private:
    Ui::MainWindow *ui;  // Вказівник на об'єкт, який управляє UI елементами

    // Допоміжна функція для додавання запису до історії обчислень
    void addToHistory(double a, double b, double result);
```

```
};

#endif // MAINWINDOW_H
```

№ Опис алгоритму створення проекту
десктоп-застосунку

Кроки створення проекту в Qt Creator:

1. Запуск Qt Creator
   Відкрити Qt Creator.
2. Створення нового проекту
   Обрати *File → New File or Project... → Application → Qt Widgets Application*. Натиснути *Choose*.
3. Налаштування проекту
   Вказати ім'я проекту і директорію для збереження. Натиснути *Next*.
4. Вибір kit (платформи для збірки)
   Вибрати компілятор та платформу, наприклад Desktop Qt 6.x.x. Натиснути *Next*.
   Створення форми
   За замовчуванням створюється клас MainWindow із головним вікном. Натиснути *Finish*.
5. Відкриття редактора інтерфейсу
   Відкрити файл mainwindow.ui — з’явиться візуальний редактор форми.

ДОДАТОК Б

Скрін-шоти вікна виконання програми та діаграми

На рис.1 показано десктоп-застосунку лабораторна робота№11

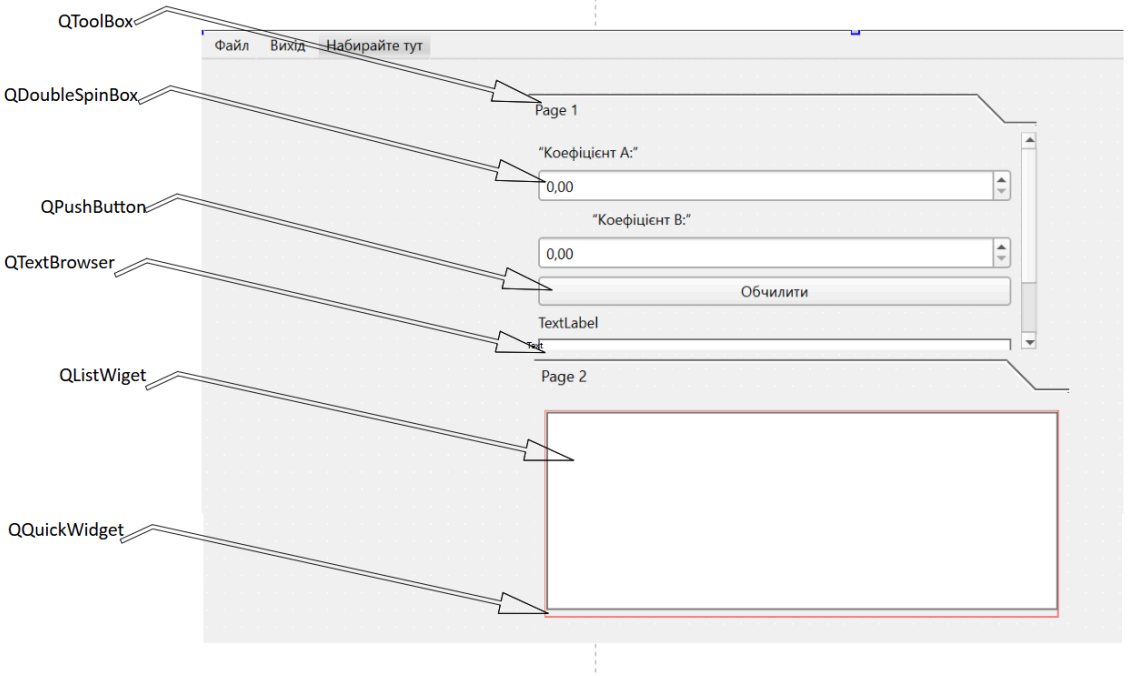

Рисунок 1 — десктоп-застосунку із 6 віджетів .

Віджети:

QToolBox

- currentIndex – індекс поточної вкладки
- count – кількість вкладок
- itemText(int index) – текст вкладки
- itemIcon(int index) – іконка вкладки
- toolTip – текст підказки
- styleSheet – стилізація
- enabled – активність

QDoubleSpinBox

- value – поточне числове значення
- minimum, maximum – межі значення
- singleStep – крок зміни
- decimals – кількість десяткових знаків

- prefix, suffix – текст перед/після числа
- readOnly – режим лише для читання
- alignment – вирівнювання вмісту

QPushButton

- text – текст на кнопці
- checkable – можливість встановити стан (натиснуто/не натиснуто)
- checked – поточний стан
- icon – іконка кнопки
- toolTip – підказка при наведенні
  enabled – доступність кнопки
- styleSheet – зовнішній вигляд

QTextBrowser

- html, plainText – вміст у форматі HTML або простого тексту
- readOnly – режим перегляду без редагування
- openExternalLinks – відкриття зовнішніх посилань у браузері
- source – шлях до зовнішнього HTML-файлу
- font – параметри шрифту
- textInteractionFlags – режим взаємодії (копіювання, виділення)
- styleSheet – стилізація

QListWidget

- count – кількість елементів у списку
- currentItem, currentRow – поточний обраний елемент
- selectionMode – режим вибору (один, множинний тощо)
- sortingEnabled – ввімкнення автоматичного сортування
- dragDropMode – перетягування елементів
- alternatingRowColors – чергування кольорів рядків

QQuickWidget

- source – шлях до QML-файлу, який завантажується

- resizeMode – спосіб зміни розміру (наприклад, SizeRootObjectToView)
- status – статус завантаження QML
- clearColor – фон компонента
- errors() – доступ до помилок компіляції QML
- focus – чи має елемент фокус

На рис.2 диаграма виконання коду лабораторна робота№11

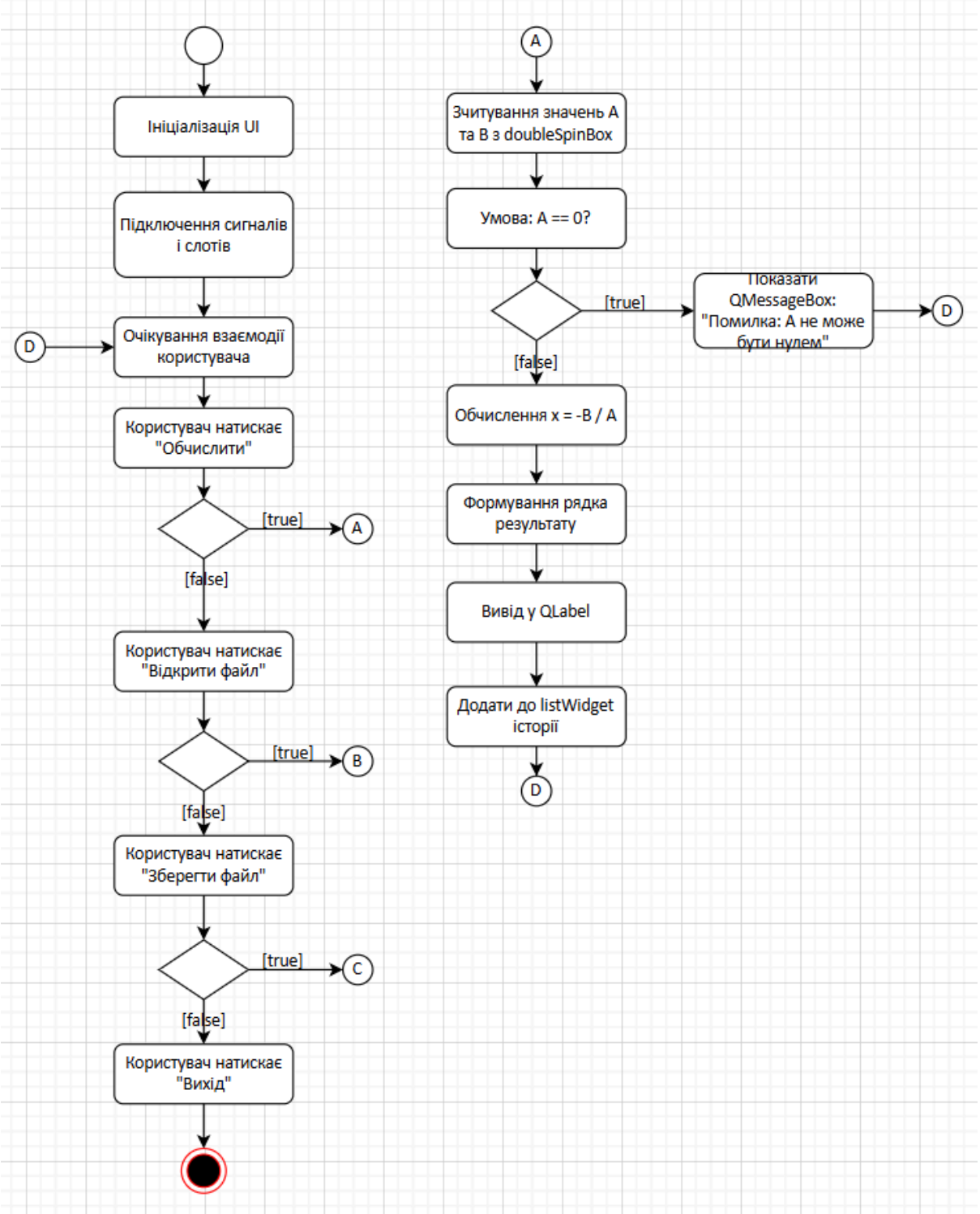

Рисунок 2 — Діаграма активності вибору меню та функція ділення

На рис.3 диаграма виконання коду лабораторна робота№11

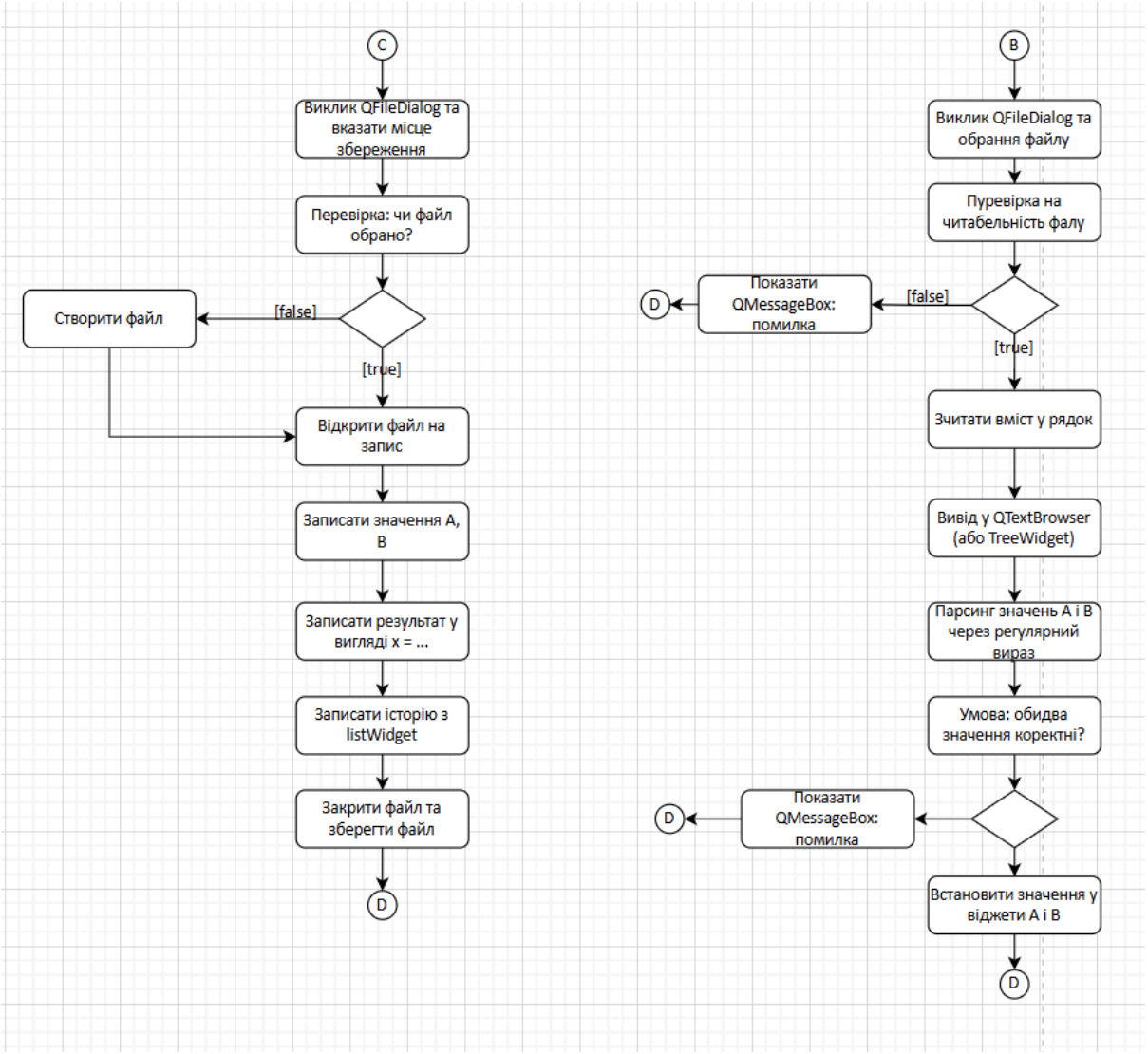

Рисунок 3 — Діаграма активності вставлення та збереження файлу